

EXHIBIT F

I N S I D E

WINDOWSTM 95

ADRIAN KING

Microsoft
PRESS

I N S I D E

WINDOWSTM 95

ADRIAN KING



PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Copyright © 1994 by Adrian King

All rights reserved. No part of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

Library of Congress Cataloging-in-Publication Data

King, Adrian, 1953-

Inside Windows 95 / Adrian King.

p. cm.

Includes index.

ISBN 1-55615-626-X

1. Windows (Computer programs) 2. Microsoft Windows (Computer file) I. Title.

QA76.76.W56K56 1994

005.4'469--dc20

93-48485

CIP

Printed and bound in the United States of America.

1 2 3 4 5 6 7 8 9 QMQM 9 8 7 6 5 4

Distributed to the book trade in Canada by Macmillan of Canada, a division of Canada Publishing Corporation.

A CIP catalogue record for this book is available from the British Library.

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office. Or contact Microsoft Press International directly at fax (206) 936-7329.

PageMaker is a registered trademark of Aldus Corporation. Apple, AppleTalk, LaserWriter, Mac, Macintosh, and TrueType are registered trademarks of Apple Computer, Inc. LANtastic is a registered trademark of Artisoft, Inc. Banyan and Vines are registered trademarks of Banyan Systems, Inc. Compaq is a registered trademark of Compaq Computer Corporation. CompuServe is a registered trademark of CompuServe, Inc. Alpha AXP, DEC, and Pathworks are trademarks of Digital Equipment Corporation. LANstep is a trademark of Hayes Microcomputer Products, Inc. HP and LaserJet are registered trademarks of Hewlett-Packard Company. Intel is a registered trademark and EtherExpress, Pentium, and SX are trademarks of Intel Corporation. COMDEX is a registered trademark of Interface Group-Nevada, Inc. AS/400, IBM, Micro Channel, OS/2, and PS/2 are registered trademarks and PC/XT is a trademark of International Business Machines Corporation. 1-2-3, Lotus, and Notes are registered trademarks of Lotus Development Corporation. Microsoft, MS, MS-DOS, and XENIX are registered trademarks and ODBC, Win32s, Windows, Windows NT, and the Windows operating system logo are trademarks of Microsoft Corporation. MIPS is a registered trademark and R4000 is a trademark of MIPS Computer Systems, Inc. NetWare and Novell are registered trademarks of Novell, Inc. Soft-Ice/W is a registered trademark of Nu-Mega Technologies, Inc. DESQview is a registered trademark and Qemm is a trademark of Quarterdeck Office Systems. OpenGL is a trademark of Silicon Graphics, Inc. PC-NFS, Sun, and Sun Microsystems are registered trademarks of Sun Microsystems, Inc. TOPS is a registered trademark of TOPS, a Sun Microsystems company. UNIX is a registered trademark of UNIX Systems Laboratories.

Acquisitions Editor: Mike Halvorson

Project Editor: Erin O'Connor

Technical Editors: Seth McEvoy and Dail Magee, Jr.



CONTENTS SUMMARY

<i>Foreword</i>	xvii
<i>Preface</i>	xxi
<i>Introduction</i>	xxv
CHAPTER ONE	
THE ROAD TO CHICAGO	1
CHAPTER TWO	
INTEL PROCESSOR ARCHITECTURE	33
CHAPTER THREE	
A TOUR OF CHICAGO	63
CHAPTER FOUR	
THE BASE SYSTEM	103
CHAPTER FIVE	
THE USER INTERFACE AND THE SHELL	157
CHAPTER SIX	
APPLICATIONS AND DEVICES	223
CHAPTER SEVEN	
THE FILESYSTEM	275
CHAPTER EIGHT	
PLUG AND PLAY	309
CHAPTER NINE	
NETWORKING	341
CHAPTER TEN	
MOBILE COMPUTING	381
EPILOGUE	
LEAVING CHICAGO	407
<i>Glossary</i>	427
<i>Index</i>	455



TABLE OF CONTENTS

<i>Foreword</i>	<i>xvii</i>
<i>Preface</i>	<i>xxi</i>
<i>Introduction</i>	<i>xxv</i>

CHAPTER ONE

THE ROAD TO CHICAGO	1
The Mission for Windows 95	3
Help for the End User	3
Hardware Platforms	4
For the Developer—32 Bits at Last	5
Shall We Go to Chicago or Cairo?	6
First Stop—Chicago	7
Clients and Servers	8
And On to Cairo	10
Summary	12
Project Goals	13
Compatibility	14
The Compatibility Fallback	15
Performance	16
Robustness—Adieu UAE?	17
Timely Product Availability	18
Easy Setup and Configuration	19
The Plug and Play Initiative	20
Configuring Windows	21
User-Level Operations	21
New Shell and User Interface	22
The New Shell	22
Complete Protected Mode Operating System	23
32-Bit Application Support	24
The Jump to 32 Bits	26
Networking and Mobile Computing	27

Bringing Windows 95 to Market	28
For Microsoft—The Bottom Line	30
Conclusion	31

CHAPTER TWO

INTEL PROCESSOR ARCHITECTURE	33
Intel Inside	34
The Intel Processor Family	35
Backward Compatibility	36
Processor Architecture	37
The 8080 and 8086 Processors	38
The 640K Barrier	39
The 80286 Processor	41
The 80386 Processor	43
80386 Memory Addressing	45
80386 Descriptor Format	45
The Descriptor in Summary	48
Virtual Memory	48
Virtual Memory Management	49
Good Virtual Memory Management	50
Mixing 286 and 386 Programs	54
The Protection System	54
Memory Protection	55
Operating System Protection	56
Device Protection	57
Low-Level Device Access	57
High-Level Device Access	58
Using the 80386 Device Protection Capabilities	59
Virtual 8086 Mode	60
Conclusion	61

CHAPTER THREE

A TOUR OF CHICAGO	63
System Overview	63
The Base System	66
Windows and Modes	67

Virtual Machines	68
Windows Virtual Machines	70
Initialization	70
The System Virtual Machine	71
MS-DOS Virtual Machines	72
Protected Mode MS-DOS Applications	73
DPMI	74
Multitasking and Scheduling	75
Multitasking Models	76
Critical Sections	79
Processes in Windows	80
Modules	80
API Support	81
Dynamic Linking	82
Support from the Base System	84
Memory Management	85
Application Virtual Memory	86
Heap Allocation	87
Windows 95 Application Memory Management	87
System Memory Management	88
Windows Device Support	90
Device Virtualization	90
Minidrivers	91
The Windows Interface	92
What Is a Window?	92
Windows 95 User Interface Design	95
Windows Programming Basics	96
Event Driven Programming	96
Message Handling	97
Program Resources	99
Windows 95 Programming	99
Conclusion	101
References	101

CHAPTER FOUR

THE BASE SYSTEM	103
Windows 95 Diagrammed	104

Windows 95 Surveyed	106
Protection Rings in Windows 95	107
Windows 95 Memory Map	108
Tasks and Processes	110
Virtual Machine Management	111
Real MS-DOS	111
Virtual Machine Scheduling	112
The Windows 95 Schedulers	114
Scheduling Within the System Virtual Machine	116
Controlling the Scheduler	116
Threads and UAEs	117
Threads and Idle Time	118
Application Message Queues	119
Physical Memory Management	121
Virtual Memory Management	125
Memory Mapped Files	127
Reserving Virtual Address Space	128
Private Heaps	129
Virtual Machine Manager Services	129
Calling Virtual Machine Manager Services	131
VMM Callbacks	131
Loading VxDs	132
The Shell VxD	134
Getting Around in Ring Zero	135
Calling Windows 95 Base OS Services	137
Calling from One VxD to Another	138
VMM Service Groups	140
Application Support	141
The API Layer	142
Mixing 16-Bit and 32-Bit Code	143
The Win32 Subsystem	147
Internal Synchronization	149
Conclusion	155
References	156

CHAPTER FIVE**THE USER INTERFACE AND THE SHELL 157**

Improving on Windows 3.0 and 3.1	159
System Configuration and Control	160
Program Manager, File Manager, Task Manager	160
Control Functions	162
Consistency	162
Visuals	164
Scalability	164
Concepts Guiding the New User Interface	165
The Document-Centric Interface	166
Look and Feel	167
The Windows 95 Shell	169
Folders and Shortcuts in the Windows 95 Shell	170
Desktop Folders	172
System Setup	173
The Initial Desktop	174
The Desktop	177
The Taskbar	179
On-Screen Appearance	182
Light Source	184
Property Sheets	185
Online Help	186
Implementation	188
Design Retrospective	189
The Outside Influences	189
The Development of the Shell	190
Changes in the Shell	192
The Taskbar	194
Folders and Browsing	195
Animation	196
The Transfer Model	197
Other Changes	198
The New Appearance	198
Screen Appearance	198
Visual Elements	201
Scalability	201

Menus	202
Window Buttons	204
Icons	204
Proportional Scroll Box and Sizing Handle	205
New Controls	205
Tool Bar Control	205
Button List Box Control	206
Status Window Control	206
Column Heading Control	207
Progress Indicator Control	208
Slider Control	208
Spin Box Control	208
Rich Text Control	209
Tab Control	209
Property Sheet Control	209
List View and Tree View Controls	210
New Dialog Boxes	210
File Open Dialog	211
Page Setup Dialog	213
Long Filenames	213
Windows 95 Support for MS-DOS Applications	215
Application Guidelines for Windows 95	217
Follow the Style Guidelines	218
Support Long Filenames	218
Support UNC Pathnames	218
Register Document and Data Types, and Support Drag and Drop	218
Use Common Dialogs	219
Reduce Multiple Instances of an Application	219
Be Consistent with the Shell	219
Revise Online Help	219
Support OLE Functionality	220
Conclusion	220
Reference	221

CHAPTER SIX**APPLICATIONS AND DEVICES 223**

The Win32 API	224
Goals for Win32	226
Components of the Win32 API	227
The Win32 API on Windows 95	229
Porting to the Win32 API	229
Porting Tools	229
API Changes	230
Memory Management	232
Version Checking	233
Nonportable APIs	233
Win32 on Windows 95	234
Security APIs	234
Console APIs	235
32-Bit Coordinate System	235
Unicode APIs	235
Server APIs	236
Printer Support	236
Service Control Manager APIs	236
Event Logging	236
Detailed Differences	237
Programming for Windows 95	238
Multitasking	238
Memory Management	241
Plug and Play Support	241
The Registry	242
The User Interface	245
OLE	245
International Support	248
Structured Exception Handling	249
The Graphics Device Interface	252
GDI Architecture	255
Performance Improvements	256
Limit Expansion	256
New Graphics Features	257

TrueType	258
Metafile Support	258
Image Color Matching	259
Color Profiles	261
Communicating Color Information	261
The Display Subsystem	262
The DIB Engine	265
The Display Mini-Driver	266
Bank-Switched Video Adapters	267
Interfacing with the DIB Engine	268
The Printing Subsystem	269
Printing Architecture	270
The Printing Process	270
Using the Universal Printer Driver	272
Conclusion	274
References	274

CHAPTER SEVEN

THE FILESYSTEM	275
Overview of the Architecture	277
Long Filename Support	281
Storing Long Filenames	282
Generating Short Filenames	288
MS-DOS Support for Long Filenames	289
Long Filenames on Other Systems	291
Installable Filesystem Manager	291
Calling a Filesystem Driver	293
Filesystem Drivers	294
FSD Entry Points	295
I/O Subsystem	296
Device Driver Initialization	298
Controlling an I/O Request	299
Calldown Chains	300
Asynchronous Driver Events	301
Interfacing to the Hardware	302
Initialization	302
Execution	303
Interrupt	303

Other Layers in the Filesystem Hierarchy	303
Volume Tracking Drivers	304
Type Specific Drivers	305
SCSI Manager	306
Real Mode Drivers	307
Conclusion	308
References	308

CHAPTER EIGHT

PLUG AND PLAY	309
Why Do We Need Another Standard?	310
History of the Plug and Play Project	312
Goals for Plug and Play	314
Easy Installation and Configuration of New Devices	315
Support for a New Hardware Standard	315
New ISA Board Standard	317
Seamless Dynamic Configuration Changes	318
Compatibility with the Installed Base and Old Peripherals	319
Operating System and Hardware Independence	320
Reduced Complexity and Increased Flexibility of Hardware	320
The Components of Plug and Play	321
How the Subsystem Fits Together	325
After a System Configuration Change	328
Hardware Tree	328
Device Nodes	329
Device Identifiers	331
Hardware Information Databases	332
Plug and Play Events	333
Configuration Manager	333
Enumerators	334
Resource Arbitrators	335
Plug and Play BIOS	336
Plug and Play Device Drivers	337
Applications in a Plug and Play System	338
Conclusion	339
References	340

CHAPTER NINE

NETWORKING	341
Windows Networking History	342
Networking Goals	346
Network Software Architecture	347
WOSA	348
Network Layers	351
Network Operations	353
The Multiple Provider Router	355
32-Bit Networking APIs	357
Network Resources	357
Connection APIs	358
Enumeration APIs	359
Error Reporting APIs	360
Local Device Name APIs	360
UNC APIs	360
Password Cache API	360
Authentication Dialog API	361
Interfacing to the Network Provider	361
The Network Provider	362
Network Provider Services	363
Device Redirection SPI	364
Shell SPI	365
Enumeration SPI	365
Authentication SPI	366
Network Transports	366
Network Device Drivers	368
Network Driver Compatibility	369
Network Configurations	370
The Network Server	372
Server Components	373
Network Printing	375
Network Security	377
Access Controls	378
Share-Level Security	379
User-Level Security	379
Conclusion	379
Reference	380

CHAPTER TEN

MOBILE COMPUTING	381
Remote Communications Support	382
Remote Network Access	385
Types of Remote Access	386
The Telephony API	389
Telephony Applications	390
Modem Support	391
The Communications Driver	392
The Info Center	394
Info Center Applications	396
Messaging APIs	396
Messaging Service Providers	397
Portable System Support	398
Power Management	398
Docking Station Support	399
File Synchronization	400
The Briefcase API	403
Conclusion	404

EPILOGUE

LEAVING CHICAGO	407
<i>Glossary</i>	<i>427</i>
<i>Index</i>	<i>455</i>

Publisher's Note

As we went to press, some aspects of Windows 95 were still under a general nondisclosure agreement, but Microsoft had made public a great deal of information about Windows 95. This book offers an interpretation of that information, and the author's conclusions are based on his exploration of Beta-1. The "Chicago" story continues to unfold, and the product will continue to be refined. For up-to-the-minute changes in information on Windows 95, we recommend that you periodically visit the WIN_NEWS forum, which you can find at the following locations:

On CompuServe: *GO WINNEWS*

On the Internet: *ftp://ftp.microsoft.com/PerOpSys/Win_News/Chicago*
http://www.microsoft.com

On AOL: keyword *WINNEWS*

On Prodigy: jumpword *WINNEWS*

On Genie: *WINNEWS* file area on Windows RTC

You can also subscribe to Microsoft's electronic newsletter *WinNews*. To subscribe, send Internet e-mail to *enews@microsoft.nwnet.com* and put the words *SUBSCRIBE WINNEWS* in the text of the e-mail.

When Windows 95 is released, be sure to head to your bookstore for complete accounts of developing for and using Windows 95.

Microsoft Press
September 16, 1994



C H A P T E R F I V E

THE USER INTERFACE AND THE SHELL

Microsoft's introduction of Windows 3.0 in New York on May 22, 1990, was the cornerstone upon which the Windows product line has built an ever increasing market share over the last few years. Although there were many notable new features in the Windows 3.0 release, the product introduction and a large proportion of the product's reviews focused on the improved visual appeal of the Windows interface. Many small, simple improvements to the interface, such as buttons that appeared to move when the user clicked them with the mouse, enhanced the product's immediate appeal—perhaps out of all proportion to their actual importance. The product's eventual success was a function of the other major new features of Windows 3.0 plus Microsoft's intense marketing campaign and the availability of some important new Windows application products. But in the first flush of the product's success, its visual appeal counted for a great deal.

Windows 95 looks as dramatically different from Windows 3.0 (and 3.1) as Windows 3.0 did from its predecessors. From the moment you start Windows 95, you can see that the appearance of Windows has been completely altered. Figures 5-1 and 5-2 on the next page illustrate the difference. Each shows one of the first screens a user sees after initial installation.

So why change a winning formula so completely? Aren't there some major business risks associated with asking a loyal base of users to accept change one more time? Of course there are some risks, and the reception of Windows 95 will determine whether Microsoft's gamble pays off.¹ In this chapter, we'll look at all the new elements of the Windows interface

1. Late in the project Microsoft decided to retain versions of the Windows 3.1 Program Manager and File Manager as desktop accessories for Windows 95—no doubt to lessen the initial shock for experienced Windows 3.1 users.

and in particular at the new Windows 95 shell—itself significantly different from the Windows 3.1 Program Manager.

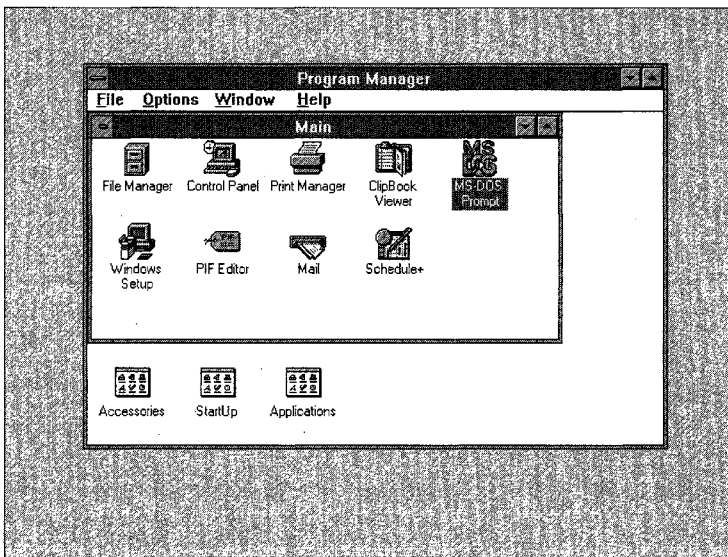


Figure 5-1.

The initial default user screen for Windows 3.1.

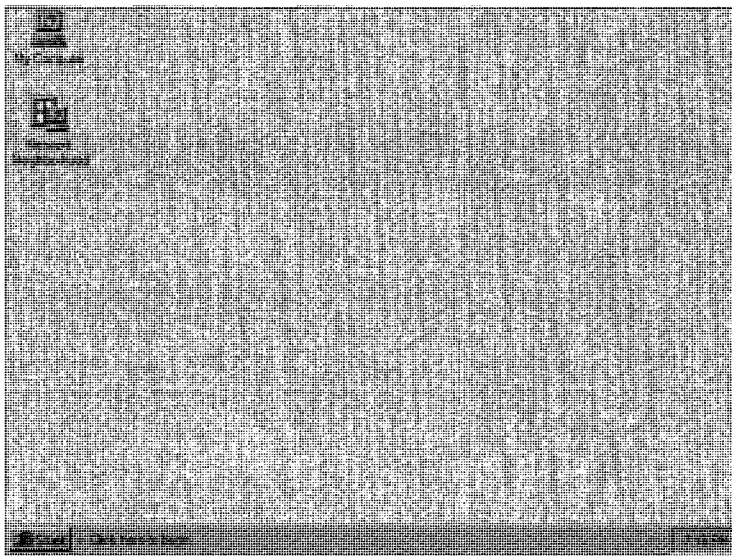


Figure 5-2.

The initial default user screen for Windows 95.

executes the switching code by running through it—not even a function call to get in its way as it comes steaming through!

Providing the bank-switching support as a standard part of the system (and making sure it runs as fast as possible) makes the mini-driver solution applicable to a much broader range of display adapters, so the likelihood of your system's using the DIB engine is pretty high.

Interfacing with the DIB Engine

When Windows 95 first loads a display mini-driver and calls the driver's DLL initialization routine, the driver simply collects information about its own configuration from the SYSTEM.INI file. Later on in the system's initialization process, GDI calls the driver's *Enable* interface twice. The first time through, the driver calls *DIB_Enable()*. The DIB engine hands back a pointer to an appropriate GDIINFO structure. The driver fills in some of the device-dependent fields (for example, the number of bits per pixel) and returns the GDIINFO structure pointer to GDI. The second call to *Enable* is where the rest of the initialization work gets done, including calling the display VxD to set the hardware into the correct graphics mode (using an INT 10) and if necessary handing the bank-switching code to the VFLATD VxD.

Once all the initialization is over, GDI, the mini-driver, the DIB engine, VFLATD, and the display VxD are all hooked together and ready to actually put something on the screen. The display mini-driver provides a standard set of about 30 or so interfaces that allow GDI to interact with the driver. Many of these functions are the same as those defined for existing Windows 3.1 display drivers, such as those for managing the cursor. All of them are exported entry points from the driver DLL. Several functions simply accept the call from GDI and hand it directly to the DIB engine. For example, GDI will call the driver's *BitmapBits()* function whenever an application creates or copies a bitmap. The mini-driver can turn around and call the DIB engine's *DIB_BitmapBits()* entry point with no transformation of parameters or, indeed, any other processing.

Management of the cursor is handled largely by the mini-driver, and, as with Windows 3.1 display drivers, the mini-driver must define the set of standard cursor resources used by GDI. This includes objects such as the standard arrow pointer, the I-beam cursor used in text fields, and the cursor we all hope we'll see a lot less of, the hourglass.